Chapter 15. Creating Documentation

Authors: *Joseph T. Buck*
Alan Kamas
Edward A. Lee

Other Contributors: *Christopher Hylands*
Phil Lapsley

Since Ptolemy is by design an extensible system, the documentation must be also be extensible. This chapter explains the document formatting conventions, scripts, and sample documents that are distributed with Ptolemy. At this time, we use a combination of text processing systems for documentation. The main ones are FrameMaker[1] and troff, although there are also some TeX components. A version of this particular chapter is distributed as a sample document in `$PTOLEMY/doc/samples/documents.book`[2]. It is made using FrameMaker, and can be used as a template for generating new FrameMaker documents in the Ptolemy style.

For users who do not have access to FrameMaker, a compatible alternative document formatting system based on troff is also provided. Currently, all documentation for stars, galaxies, and demo programs is based on troff. All shell scripts and Makefiles are supplied along with the documentation so that they can be modified if necessary. The formatting is with `ditroff` (device independent troff) with the -me macros, `eqn`, and `tbl`. In addition, some custom macros have been supplied in order to enforce a consistent look across all Ptolemy documents. These macros and their use are explained below. The -me macros will not be covered, since they are adequately documented elsewhere. Troff documents created in this way can be converted to FrameMaker documents using the script `$PTOLEMY/doc/fmconvert/nr2mml`. This script creates an "mml" file that can be imported into FrameMaker. The conversion is not perfect, however, so some manual editing will be required. In addition, the script `$PTOLEMY/doc/fmconvert/extractSD` can be used to extract the descriptors of stars, galaxies, and demos to generate FrameMaker documents like those found in the individual domain chapters of this manual.

15.1 Printing the manual

The simplest way to get a hard copy version of the manual is to have a double sided bound copy sent to you. You may order the documentation set from:

EECS/ERL Industrial Liaison Program Office, Software Distribution
205 Cory Hall
University of California at Berkeley

1. FrameMaker is a registered trademark of the Frame Technology Corporation.
2. $PTOLEMY is an environment variable that is assumed to specify the installation directory for the Ptolemy system.

Berkeley, CA 94720
phone: (510) 643-6687
fax: (510) 643-6694
email: ilpsoftware@eecs.berkeley.edu

If you would like to print out your own copy of the documentation, you will need a postscript printer. All of the Ptolemy documentation is contained in a collection of postscript files. These files have the ".ps' suffix at the end of their file names. The files are found in the Ptolemy distribution as follows:

`$PTOLEMY/doc` contains most of the documentation. Within `$PTOLEMY/doc`, the directory `users_man` contains the Ptolemy user's manual. The directory `prog_man` contains the programmer's manual. The `bin` directory contains scripts for building and printing troff based documentation. The `headers` directory contains troff header files that are needed by the scripts in the `bin` directory. The `main` directory contains the makefiles needed to print out troff based documents. Finally, the directory `$PTOLEMY/doc/samples` contains sample documents and templates to follow if you are planning to add new documentation.

Domains and their stars are documented where their source code resides. For instance, the documentation for the SDF domain is in `$PTOLEMY/src/domains/sdf/doc`. The documentation for each of the stars is kept in the `stars` subdirectory, and the documentation for each demo is in the `demos` subdirectory. A star documentation file is automatically added whenever you create a new star. The format for this documentation, and methods for printing it are found in the "Documenting Stars" sub-section in the "Writing new sections using troff" on page 15-5.

Each of the documentation directories mentioned above may have a `README` file that explains which postscript files are which and explains how to print out the files. In any event, you can print the section you are interested in by going to the documentation directory and then printing out the postscript files found there.

15.2 Using FrameMaker

In the directory `$PTOLEMY/doc/samples`, the following files can be found:

`title.doc`:	A sample cover page.
`documents.doc`:	A sample document (very similar to this chapter).
`documentsTOC.doc`:	The table of contents for the document.
`documents.book`:	A book file that unites the above three documents.

These documents can be used as models or templates for creating new documents that are to be inserted in *The Almagest* or are to stand alone. Use documents.doc as a template for most applications. It defines the paragraph and character styles visible in this chapter.

By convention, except for the sample document, we do not distribute the FrameMaker files for the entire *Almagest*. Instead, we distribute the PostScript[1] code produced by Frame. The makefiles used to print manual, therefore, simply assume that the PostScript files are up to

1. PostScript is a registered trademark of Adobe Systems Inc.

date. It is up to you to ensure this. You must also ensure that the index files corresponding to the PostScript code are up to date. The section below explains how to generate these.

15.2.1 Index Entries

Since Ptolemy documentation is generated using a variety of word processors, the index must be generated with some care to ensure consistency. In particular, FrameMaker documents should not just generate their own, stand-alone index. The indices will be merged when you print the entire Almagest, or when you invoke "make prindex" in $PTOLEMY/doc. Thus, when creating a FrameMaker document, index markers are used in a non-standard way.

The following X resources should be modified in the file `$FMHOME/fminit/usenglish/Maker.us`:

```
Maker*marker.10: IndexReference
Maker*marker.11: IndexExamplle
Maker*marker.12: IndexDefinition
Maker*marker.13: IndexStarRef
Maker*marker.14: IndexStarEx
Maker*marker.15: IndexStarDef
```

These resources cause the named index markers to appear in the list of markers.

To make an index entry in a FrameMaker document, select the text you wish to appear in the index, and select the FrameMaker command Special-Marker (Esc-s-m). Then choose one of the above six types of index entries, using the following guidelines:

`IndexReference`: Generic index entry

`IndexExample`: An example of the usage of a particular feature.

`IndexDefinition`: The definition of a term.

`IndexStarRef`: A generic reference to a star.

`IndexStarEx`: An example of the usage of a star. For example if the text that describes the SDF `butterfly` demo would have a index entry that looks like: `butterfly (SDF demo)`.

`IndexStarDef`: The definition of a star. This entry is normally automatically generated when a star is compiled, so you will probably not encounter any occasion to use it directly. The text that defines the SDF `Ramp` star would have the marker text: `Ramp (SDF block)`

Avoid index entries beginning with very generic words in the Ptolemy vocabulary, like Ptolemy, `star`, `galaxy`, or `domain`. Of course, if you are writing some explanation of these basic terms, then an index entry is appropriate. Before entering index entries for a star, look in the documentation for similar stars to get an idea of the subject terms that have already been used and might be related. Be sure to follow the same capitalization rules as the existing index entries (i.e. `Ramp (SDF block)`, not `Ramp (SDF Block)`).

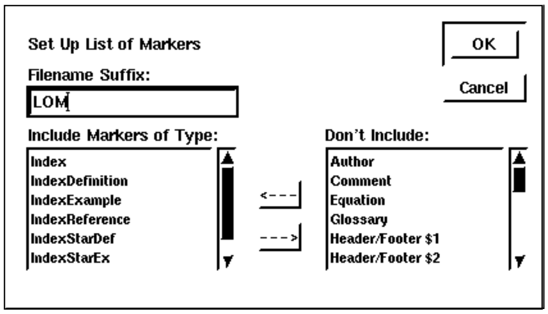
Currently we use Quadralay's WebWorks to convert Framemaker documents to html. You can use marker 24 to indicate text that should be present in the html output as a link. Use a marker 25 on the end of the text of the text that represents the link. The text of the first marker 24 contains the filename or URL:

`http://ptolemy.eecs.berkeley.edu`

The marker 25 is placed at the end of the text you want underlined in the html version.Here is a sample link to `http://ptolemy.eecs.berkeley.edu`.

The markers are not printed when the FrameMaker document is printed. WebMaker converts the text delimited by the markers into HTML hypertext links.

When you print your document, you should generate the index file that will be used to print the overall index. To do this, select File/Generate, and within the ensuing dialog box, select List/List of Markers. In the dialog box that results from this, be sure all of the above index markers are included, and then accept the default filename suffix “LOM”. The dialog box should look like this:



When you click OK, you will get a new file with a list of markers in a format acceptable to the Ptolemy index generation software. This file should be saved in “Text only” form. By convention, we name the index file using the document name with the suffix “.index”.

15.2.2 Special fonts and displays

- By convention, Ptolemy documentation uses a special font for C++ class names. In the FrameMaker template, the corresponding format is named `Class`.

```
For a display entirely set in this font,
use the "Commands" paragraph format, as shown here.
You can use "meta-Return" to force carriage returns where
you want them without getting a new paragraph.
```

- Note that if you use the Commands paragraph format, you should be sure to change any slanted double quotes “ ” to straight quotes `"`, by typing Control-Shift-`"`. In general, commands don’t have slanted double quotes, hence the need to convert them to straight quotes.
- Star and Target parameters, such as the SDF *loopScheduler* target parameter, are always in italic, use the Emphasis font format.
- The names of stars and demos should be in the `Class` format.
- If a string is to be taken literally, it should be in the `ProgramCode` character format. An example would be that the default of the *loopScheduler* target parameter is `1`. Strings such as `YES`, `NO`, `TRUE` and `FALSE`, that are used as values to parameters should also be in the `ProgramCode` format.
- The first line of a dialog with a computer should use the `Commands` paragraph format.

Each successive line should use the `CommandsCont` paragraph format. The text the computer would print should be in the `ProgramCode` character format, the text the user would type should be in the **`ProgramUser`** character format.

- If a string is only an example and to be substituted with a proper value, it should be in *`ProgramVariable`* character format.
- To include images of palettes in documents, see “Capturing a screen image” on page 2-41.

15.3 Writing new sections using troff

A set of troff macros is used to control most important formatting options. These macros are explained here. The set of macros used to document basic capabilities is different from the set used to document demos, stars, and targets. For stars, the formatted documentation will be generated automatically by the preprocessor. For demos and targets, you must generate it yourself.

15.3.1 Basic documents

The macros used in troff documents are defined in `$PTOLEMY/doc/headers`. In that directory, macros are defined in the file called `shared`, and in either the file called `standalone` or `together`, depending on whether the document is printed by itself or in combination with other manual sections.

A typical document has the following syntax:

```
.\" %W% %G%
.VR 0.%I% %G%
.TI "SECTION TITLE"
.AU
Author list
one name per line
no punctuation
.AE
.H1 "Section Title"
.H2 "Title"
.H3 "Title"
.H4 "Title"
.UH "Un-numbered header"
```

Text appearing between headers uses standard -me macro formatting. The first line is a comment that will be translated by SCCS into a version number for the document, looking something like:

```
.\" @(#)filename 1.1 10/14/93
```

This is interpreted by `ditroff` as a comment, and hence has no affect on the output. The version number that will appear on the output is specified by the .VR macro. It takes two arguments, a version number and a date. These two arguments will be printed in the lower left and right corners of each page, respectively. If the version number is given as `%I%`, it will be translated by SCCS into the same version number that follows `filename` in the `@(#)` line above. If the date is given by `%G%`, it will be translated by SCCS into a date as shown in the lower right hand corner of this document.

The `.TI` macro gives the section title. When the document is printed standalone, this title will be put on a separate title page. Otherwise, it will be a level-one header. The rest of the macros should be self explanatory. A table of contents will be generated using these headers.

When printing a troff document, you can pass options to *ditroff* through make. For instance, suppose you wish to print only pages 48 through the end of section 1. Then use

```
make OPTIONS=-o48- print
```

Any set of `ditroff` options may be passed this way.

15.3.2 Adding new manual sections

Your new directory should contain a file called `overall` which defines the entire document. It may contain lines like

```
.so filename
```

to source from another file. Typically, all major subsections of the document would be written in their own file, with a corresponding `.so` entry in `overall`.

Your new directory should also itself contain a `Makefile`, used for printing the document in standalone form. This should be a symbolic link to the file `$PTOLEMY/doc/main/sectionMake` — using a link rather than a copy guarantees that all sections will print the same even if the master copy is modified.

15.3.3 Index entries

The documentation scripts for Ptolemy will automatically generate an index. The indexing system supports three kinds of index entries: "definition" entries are used when a term is defined in some sense; "example" entries are used when a term is used in an example; and "reference" entries are used in all other cases — typically when a term is referenced. When the index is printed, definition entries have their page numbers in boldface, example entries have their page numbers in italics, and reference entries have their page numbers in the default font. This allows the reader to quickly locate the information for which they are searching.

Entries in the index must be defined in the troff document source using one of the macros:

```
.Id "definition entry"
.Ie "example entry"
.Ir "reference entry"
```

for a definition, example, or reference entry, respectively. (The .IE macro used in older versions of Ptolemy acts as a reference entry). The entries will be sorted alphabetically. If you look at the index printed with this document, you will find an entry "index" pointing right here. If you wish for a two word entry to appear twice, alphabetically by each of the two words, then you must use two index entries:

```
.Ir "two words"
.Ir "words, two"
```

No additional editing will be done on index entries. For this reason, index entries should be carefully chosen. For example, please do not put the following entry in every demo description:

```
.IE "Ptolemy demo"
```

This will lead to hundreds of identical, and very uninteresting index entries.

One final note: to generate an index entry for a star or galaxy, use one of the macros:

```
.Sr star domain
.Se star domain
```

These will create reference or example index entry for the named star in the given domain in the "C++ class font" (to set it off from other entries), and will parenthetically note that the entry is for a star in the specified domain. If the star is in the same domain as the current section of the documentation, you can leave off the domain specifier. For example, the entry:

```
.Sr autocor
```

when in SDF domain documentation will generate an index entry of the form:

```
autocor (SDF star)
```

15.3.4 Special fonts and displays

By convention, Ptolemy documentation always refers to Ptolemy using the troff string invoked by the sequence "\Ptolemy". This gets formatted in a standard way with a standard font. The overall Ptolemy version number is 0.6, invoked by the sequence "*(PV".

In addition, c++ class names are set in a special font by using the ".c" macro:

```
.c Class
```

as shown here: `Class`. For a display entirely set in this font, with no fill, use the macro pair ".(c ... .)c", as in

```
.(c
text
goes
here.
.)c
```

which will come out like:

```
text goes here.
```

This font is also used to highlight keywords, and to display sections of source code. The ".(c ... .)c" macro pair attempt to keep the text between them on one page, starting a new page if there isn't room on the current one. The ".(d ... .)d" macro pair have the same function but without the keep.

15.3.5 Documenting stars

Manual entries for new stars will be generated automatically if you follow the directions given in the programmers manual for defining new stars. You need not, therefore, be concerned with the troff macros used for this. All editing of the star documentation should be done in the .pl file, not in the troff file.

15.3.6 Documenting new galaxies and universes

Unlike stars, galaxies and universes must be documented manually. The troff source files for galaxies should go in *XXX*/doc/stars, where *XXX* is the location of the domain source directory. Since galaxies can be used exactly as if they were stars, it is logical that their documentation should appear with the stars. Troff source files for demo universes should go in *XXX*/doc/demos, where *XXX* is again the location of the domain source directory.

In both cases, the syntax of the document source files is as follows:

```
.\" %W%%G%
.NA "Name of demo universe"
.SD
Short descriptor
.SE
.SV %I% %G%
.AL "Author list, with commas, and a period."
.LO "Location of the galaxy or demo definition"
.LD
Long descriptor.
This can contain any -me, eqn, or tbl directives.
It can also contain un-numbered headers:
.UH "title"
.SA
"See Also"
List of pointers to other entries in the manual
.ES \" Every entry must end with this call
```

The first line is again a `ditroff` comment, containing the SCCS version ID. The second line defines how the version ID and date should be printed when the manual is printed. The version number `%I%` and date `%G%` keywords have the same meaning as before. The date here must be manually modified when the galaxy or demo is modified. The rest of the entries should be self-explanatory. Here is an example:

```
.\" %W% %G%
.NA adaptFilter
.SD
A demo of the LMS adaptive filter star
.DE
.Se LMS
.SV %I% %G%
.AL "Edward A. Lee"
.LO "$PTOLEMY/src/domains/sdf/demo"
.LD
A Gaussian white noise signal serves as input to an
FIR filter and an adaptive filter using the LMS
algorithm. The output of the two filters and the error are
displayed after the run. By examining the parameters of
the two filters, you can find the file names for the FIR
filter coefficients and the initial tap values. The final
tap values can be saved by specifying a saveTapsFile in
the adaptive filter star.
.SA
IIDGaussian,
LMS,
```

```
FIR.
.ES
```

Note that a carefully chosen index entry has been inserted here via the ".Se" macro. *It is not necessary to insert an explicit index entry for the name of the galaxy or demo*. This will be done automatically. It is also recommended that index entries not be inserted for all the "See Also" entries. In this case, however, the main purpose of the demo is to demonstrate the `LMS` star, and the name of the demo is not something like `LMSdemo`, so an explicit entry is advised.

For galaxies, additional entries are required to document the inputs, outputs, and states. The list of inputs is surrounded by the delimiters

```
.IH
...
.PE
```

while the list of outputs is inside

```
.OH
...
.PE
```

Each input or output is listed as follows

```
.NE
\fIname\fR (type): descriptor
```

States are delimited by

```
.SH
...
.ET
```

with each entry reading

```
.NE
\fIname\fR (type): descriptor
.DF "default value"
```

State entries should also be made for demos with top-level parameters.

A single star, galaxy, or demo document page can be printed using the command `printpage` *`filename`*. This command is stored in `$PTOLEMY/doc/bin`, but also has a link in `$PTOLEMY/bin`, so it should be in your path.

15.3.7 Configuring your system to use the Gnu groff program

If you do not have `ditroff`, you may be able to preview and print the documents by configuring your system to use `groff` instead. The `groff` package is a set of tools from the Free Software Foundation that provide a freely redistributable, mostly-compatible replacement for `troff`, `nroff`, and related programs. Note that `groff` is not a replacement for `ditroff` and the scripts below are not guaranteed to work.

Since the command line interface to `groff` is quite different and some adjustments are required, different scripts are provided. To install them as the official document-printing scripts, do the following (you must have appropriate write permission):

```
% cd $PTOLEMY/doc/bin
```

```
% USE_GROFF
```

This command will install the `groff` scripts as the official scripts.

Since `groff` is a replacement for the old `troff` rather than for `ditroff`, it cannot handle as many fonts; therefore the scripts for using `groff` cause some of the fancier fonts used in the manuals to print in bold face instead. There may be other glitches as well; however, the resulting manuals are still quite usable.

The `groff` program can be obtained from many FTP sites, including `ftp://prep.ai.mit.edu/pub/gnu`; the Free Software Foundation will also send a copy of it to you on tape for a reasonable fee. Write to: Free Software Foundation, Inc. 675 Mass Ave, Cambridge, MA 02139, USA or send email to `gnu@prep.ai.mit.edu` for more information on obtaining FSF software.